

DARPA Research Panel 1: Secure Systems and Access Control

Panel Chair: Teresa F. Lunt, DARPA

Panelists:

Dan Sterne, TIS
Roshan Thomas, ORA
Mary Ellen Zurko, OSF
Jay Lepreau, University of Utah
John Rushby, SRI International

Over the past two decades, much of the research in computer security has been sponsored by the DoD and has focused on multilevel security (MLS). Several laboratory prototypes were built to demonstrate the feasibility of high-assurance MLS systems. However, very little of this work has transitioned. While many vendors did produce MLS versions of their products, these generally diverged from the standard products. This divergence leads users to prefer the standard versions, since most of the popular applications will not be available or may not work correctly on the lesser-known MLS versions. And this, in turn, means that those users who need MLS still do not have an affordable solution; much customization and special-purpose applications and integration code must be written.

Instead, what is desired is for vendors to build security into their mainstream products. This is feasible only if a large segment of users want the security. The security features of general-purpose products must meet the needs of a broad set of interests, not just MLS. Policy-neutral security mechanisms could enforce any number of organization-specific policies, including MLS, but would not have any single policy “wired in.” These mechanisms should allow a broad enough set of policies to be specified and enforced so as to appeal to a wide set of user communities, such as finance, health care, and commerce, as well as defense. One can envision a future in which national-security-“blessed” policies will be available from third-party vendors for use with these generic, but specializable, products.

Most organizations have more complicated information protection needs that simple mandatory and discretionary access control matrix-oriented policies are capable of expressing. In addition to the familiar mandatory and discretionary access control policies, we should also explore richer policies such as role-based, task-based, and

workflow-based policies so as to appeal to the broadest possible constituency. To do this, we need to identify a desirable range or class of policies, investigate natural ways of expressing such policies, identify and develop a common set of mechanisms capable of enforcing the desired range of policies, develop policy “compilers” to map user-specified policies into the base mechanisms, and address the related assurance issues.

Based on current research trends in operating systems, we expect future systems to be more modular. This may also be true someday of database systems. This will give us the opportunity to make security a modular and reusable component of systems. This has the advantage that the end user need only use the security modules if they need and are willing to pay for the security. It also means that various degrees of security can be made available for use with the same products. Moreover, it may be possible for several different systems to share the same security “modules,” so that a common security policy can be enforced across diverse system components. There is the additional advantage that security modules can be replaced by high-assurance national-policy-enforcing modules when the systems are used in certain defense applications.

The panelists explore these and other issues being investigated in the DARPA research program.

Domain and Type Enforcement Firewalls

Dan Sterne, TIS

The pervasive need for E-mail and world wide web services and the growing importance of electronic commerce have driven many organizations to connect their local area networks (LANs) to the Internet in spite of the significant security risks this incurs. As a defense, many organizations use firewalls to constrain interactions with the Internet, allowing only the use of those services and protocols deemed relatively safe. While firewalls are a valuable tool, they reduce but do not eliminate Internet security risks. For example, a firewall that permits outgoing E-mail cannot tell whether such E-mail contains the announcement of a company picnic or the minutes of a highly proprietary corporate strategy session. Consequently, it allows either to flow out to the Internet, indiscriminantly. Similarly, a firewall that permits LAN users to surf and view anonymous remote web sites freely will not protect LAN hosts from attack by malicious web pages containing executable content, e.g., postscript or Java.

Addressing these Internet security problems requires protection mechanisms beyond those provided by firewalls, namely, operating system security mechanisms. Unfortunately, mainstream UNIX systems (and other mainstream operating systems) provide only weak, discretionary protection mechanisms that are insufficient for these

purposes. In addition, UNIX systems are relatively easy to penetrate. In part, this is because they are difficult to configure securely, even by expert administrators. Moreover, they rely on a large number of complex programs that execute with root privilege; an attacker that subverts a single root program gains control over an entire UNIX system. Multilevel secure operating systems provide stronger protection but are viewed by many organizations as inflexible and ill-suited to the security problems of the commercial world.

Under DARPA funding, TIS is developing an integrated approach for Internet security that combines both firewall and secure operating system technologies. The foundation of this approach is a previously developed UNIX prototype whose kernel provides Domain and Type Enforcement (DTE), an extended version of Bobert and Kain's Type Enforcement. DTE is a strong yet flexible form of access control that can be configured to support a variety of site-specific security policies. An administrator configures a DTE system by writing high level access control rules in DTEL, human-friendly, machine-interpretable policy language. DTE controls access not only to files and devices, but network communications services. In order for a process on a DTE system to be able to send or receive network traffic, the traffic must be labeled with a type that is specified in the DTE policy as sendable or receivable in the process's domain. The DTE prototype currently uses the option space in IP headers to store type labels and other DTE security attributes.

The other central component in this approach is a firewall that integrates DTE and the TIS Firewall Toolkit. DTE is used in the firewall in two ways. First, the firewall is made stronger by organizing the firewall operating system components and firewall application proxies into small DTE-enforced execution domains. This increases the firewall's resistance to penetration by an attacker. Second, the firewall is made smarter by incorporating into it cognizance of the DTE capabilities and DTE policies of hosts on the LAN it protects.

In this approach, the DTE firewall's role is to support local hosts' DTE policies and to coordinate its actions, including policy updates, with other DTE firewalls. These notions are being investigated via three phases of prototyping. In the first phase, a DTE firewall attaches DTE attributes to inbound traffic from the Internet and checks the appropriateness of labeled outbound traffic. It also selectively channels to DTE hosts important but potentially dangerous network services (e.g., Java) that may convey too much security risk for ordinary (i.e., non-DTE) hosts that are also present on the LAN. In the second, two distinct enclaves protected by DTE firewalls exchange cryptographically protected network traffic. This traffic includes DTE security attributes having semantics that have been mutually agreed upon by both enclaves. This allows role-based and other kinds of security policies supported by DTE to extend across the Internet to enclaves operated by different organizations. In this phase, the DTE policies of the enclaves protected by the DTE firewalls will

differ but overlap. The policy overlap, specified in DTEL, defines the kinds of information that both enclave's owners have agreed to exchange. In the third prototype, Domain and Type Authority (DTA) Servers will provide directory-like network services so that firewalls can dynamically discover the types of information that can be exchanged safely with other firewalls.

The increased reliance of commercial and government sectors on the Internet and its associated technologies intensifies the need for improved Internet security. While firewalls and secure operating systems have critical roles to play, a comprehensive approach requires both. By combining these technologies synergistically, we hope to better address the growing security needs of the government and commercial sectors and enable the safe exchange of a broader array of services over the Internet.

Task-based Authorizations: A Research Project in Next-generation Active Security Models

Roshan Thomas, ORA

In this project, we develop a new paradigm for access control and security models, called task-based authorizations. TBA is particularly suited for emerging models of computing. In particular, this includes distributed computing and information processing activities with multiple points of access, control, and decision making. TBA articulates security issues at the application and enterprise level. As such, it takes a "task-oriented" or "transaction-oriented" perspective rather than the traditional subject-object one. Access mediation now involves authorizations at various points during the completion of tasks in accordance with some application logic. In contrast, the subject-object view typically divorces access mediation from the larger context in which a subject performs an operation on an object. By taking a task-oriented view of access control and authorizations, TBA lays the foundation for research into a new breed of "active" security models.

In a task-based approach to security, the basic entities are:

- Tasks and sub-tasks: these represent strands of activity.
- Authorizations: these are approval steps that occur at one more points in the lifetime of various tasks and sub-tasks.
- Dependencies: these are relations between authorizations and their encompassing tasks.
- Authorization policies: these are authorizations and dependencies combined to form meaningful expressions of authorization policies.

Central to the TBA approach is the notion of an authorization-step, representing a primitive authorization act. In the paper-based forms environment, the analog

of an authorization-step would be an approval on a form, identified by a signature. The active aspects of the model can be attributed to the fact that TBA recognizes the interaction of authorizations and permissions as it occurs within the lifetimes of tasks and activities, thereby enabling it to take an active role in the management of authorizations and corresponding permissions.

The key research directions that we are investigating during the course of this project include the following

- TBA as an active security model
- modeling and specification of authorization policies
- use of visual languages to specify authorization requirements and policies
- application of TBA to distributed computing and workflows

A model such as TBA can be used to address the gap that exists today between the enterprise and systems perspectives of security. Thus TBA can form a bridge between high-level enterprise security models and low-level access control models. TBA will have broad applicability in areas such as the automation of mission critical command and control scenarios where authorization sequences need to be carefully controlled, security management of complex operations in high-assurance client-server environments, as well as in forms-based workflow applications such as logistics management, distributed planning and claims processing.

User-centered Security and Adage

Mary Ellen Zurko, OSF

While "user-friendly security" is viewed as a humorous oxymoron in some circles, the security community has long acknowledged the importance of usable secure systems. There was a pragmatic recognition that secure systems that are difficult to use will get circumvented or insecurely managed by their users. In 1975, Saltzer and Schroeder identified psychological acceptability as one of eight design principles for computer protection mechanisms [2]. While other principles from that paper such as least privilege and fail-safe defaults have become standards in the security literature, there has been very little work done on user-friendly security. The lack of work in this area is due in part to the history of research, development, and use of secure systems. Most research and development in secure systems has strong roots in the military. People in the military are selected and trained to follow rules and procedures precisely, no matter how onerous. This user training and selection decreased the pressure on early secure systems to be user friendly. In another example of military influence, the first security model to achieve widespread attention in the security literature (Bell and LaPadula [1]) encoded military classification levels and need-to-know categories.

Much effort was then spent trying to apply this model to all uses of secure systems. Finally, mathematical rigor has been emphasized over usability in many security modeling efforts.

In considering how best to integrate usability and security, we considered three different approaches. We can apply established procedures for enhancing usability to developing or existing secure systems. While this approach seems the most obvious and the cheapest, it has rarely been documented. A second approach is to integrate appropriate security services into software with a strong usability component, such as mass-market applications or groupware. Most of the work in this area has focused on privacy, and has taken place in the Computer Human Interface (CHI) community. We call the third approach user-centered security[4]. The term refers to security models, mechanisms, systems, and software that have usability as a primary motivation or goal. This approach provides the tightest integration between usability and security. The timing seems right for a renewal of interest in synthesizing usability and security. There is increasing pressure on government funded researchers to produce results that can be used to solve real world problem, and the standard for ease-of-use in commercial products continues to rise.

We are pursuing our vision of user-centered security in the Adage project (Authorization for Distributed Applications and Groups) [3]. Adage will provide a toolkit that will allow distributed applications to take advantage of Adage's services, encouraging consistent mechanisms and policies throughout an organization. Adage is specifically conceived to overcome the usability problems with authorization mechanisms for distributed applications in use today.

The first of these usability problems is that the applications unnecessarily export the underlying data structure as the user model. The user metaphor for Access Control Lists (ACLs) is the ACL data structure; for system masks it is the system mask. The user is given a rudimentary formatted display of the information in the data structure (or perhaps just a literal display of its values) and must learn the algorithm that the computer software will use to evaluate that data structure in order to understand what access control policy is actually instantiated. A large gap remains between these traditional security mechanisms and a user's or site's security policy, stated in natural language. By analogy, ACLs are the assembly language of security policy. They are a complex, low-level language. Only an expert in a particular implementation of ACLs can hope to program it correctly the first time. ACLs have the added disadvantage of being difficult to test without making changes on a live system. One component of Adage will be a higher-level authorization language that begins to close the gap between security mechanisms and site security policies. It will come with a visual builder that allows site security administrators to build up an authorization policy from visible policy pieces. Furthermore, these policies can be shared with other domains. The primitives supported by this language will support

a wide range of user and application policies, because they will be based on security policies actually in use and on interviews with security administrators.

One insight that Adage shares with current work on roles is that within organizations it is natural to think about both users and objects in terms of how they relate to each other and what place they fill within the organizational structure. Adage will use groupings to reflect these intuitions. It will use groupings of objects and of actions to more easily refer to objects and actions in a security policy. Groups of users and their roles will receive particular attention. Adage will provide an infrastructure for defining the relationships and restrictions on groups and roles that will allow it to support models from both the security and groupware literature. For example, two groups can be restricted to have no membership overlap, to support static separation of duty. Users taking on the role of Chair can be restricted to those users in a particular group.

Adage will continue the work in user-centered trust models by modeling common trust dimensions such as amount of trust (How much do I trust you? How much do I distrust you?) and type of trust (What do I trust you for?). Adage will apply this trust model to services whose information is used as input to authorization decisions (such as authentication servers and group membership servers). This will allow an enterprise to articulate a trust policy and have it apply to all its authorization decisions. In addition, the model will allow trusted services to introduce other trusted services, forming chains of trust where the amount of trust degrades over hops, much as real-life trust does.

[1] Bell, D. E. and L. J. LaPadula. Secure Computer Systems: Unified Exposition and Multics, Technical Report ESD-TR-75-306, The MITRE Corp., March 1976.

[2] Saltzer, Jerome H. and Michael D. Schroeder. “The Protection of Information in Computer Systems”, in *Proceedings of the IEEE*, 63(9), 1975.

[3] Zurko, Mary Ellen. Adage home page, <http://www.osf.org/www/adage/index.html>.

[4] Zurko, Mary Ellen and Rich Simon. “User-Centered Security”, in *Proceedings of New Security Paradigms Workshop*, 1996.

Encapsulated Environments Using the Flux Operating System

Jay Lepreau, University of Utah

Most modern operating systems provide a concept of “virtual machines” — e.g., processes or tasks — and allow several such virtual machines to coexist on a single machine and compete with each other for hardware resources. Such separate processes are a classic way to support separate information domains. In the 1970’s the term “virtual machine” usually referred to an OS architecture that exported what appeared to be the naked hardware, and an entirely separate copy of a stand-alone operating

system ran on that “virtual machine.”

Based on a synthesis of microkernel and virtual machine concepts, we have developed an OS architecture that allows recursive virtual machines (virtual machines running on other virtual machines) to be efficiently implemented, in software, by a microkernel running on generic hardware. The model can also be called a “nested process model,” in which *any* process can completely contain and control other processes within it.

Virtual machines were a classic way to provide high security subsystems, fully isolated from one another. Our recursive model takes this a step further, efficiently providing *hierarchical* control by any process in the system. Such flexible and hierarchical control is ideally suited to supporting the security requirements of arbitrary untrusted applications, often loaded over the Internet and Web. Each security manager can completely control the resource (memory, cpu, higher-level services) of its children. Each child may, if it wants, implement similar control over its children. In this manner the children can control and isolate further untrusted applications.